## CS109A Solutions for Class Quiz #2

March 8, 1995

**Problem 1:**
We need to choose a set of 2 teams for each match. The order of selection doesn't matter. Therefore, the number of possible matches is $\binom{22}{2} = 231$.

**Problem 2:**
There are 9 coins: 3 nickels, 2 dimes, and 4 quarters. Since the coins of each denomination are indistinguishable, by Formula (4.7) in FCS, we can arrange these 9 coins in $9!/3!2!4!$ ways.

**Problem 3:**
There are two straightforward ways to approach this problem.
You may use as your probability space either ordered lists of 4 cards or unordered sets of 4 cards.
A choice that was not handled properly was to choose lists of 4 ranks as the points.
The problem is that these points are not of equal probability. For example, AAAA can occur only $4! = 24$ ways, corresponding to order of the suits, while A123 can occur $4^4 = 256$ ways, corresponding to selection of suits independently.
Technically, you can use this probability space, but you must then sum the probabilities of the points in your event, rather than take a ratio of counts.
Anyway, here is method 1, ordered selections:
**a.** The points of the space are the lists of 4 cards out of 52.
**b.** There are $\Pi(52, 4) = 52 \times 51 \times 50 \times 49 = 6{,}497{,}400$ points.
**c.** We can choose the first card in 52 ways. There are then 48 cards of the 12 different ranks left for the second card, 44 for the 3rd card and 40 for the 4th card, or $52 \times 48 \times 44 \times 40 = 4{,}392{,}960$.
**d.** The ratio of (**c**) to (**b**) is approximately 67.6%.
Now, here is the same problem worked without considering order.
**a.** The points in the space are sets of 4 cards out of 52.
**b.** There are $\binom{52}{4}$ or 270,725 points.
**c.** We can choose the ranks in $\binom{13}{4}$ ways, and for each rank chosen, we can pick the suit in 4 ways. There are thus $4^4\binom{13}{4} = 183{,}040$ points in the event.
**d.** Same answer, of course.
Some common errors and their keys:
X3A: Rank/Suit confusion. ($-3$). This mistake lead to an answer of 10.5%, which is the correct probability that all four cards are of different suits.
X3B: Forgetting in part (c) that suits have to be selected as well as ranks. ($-4$). This led to an answer low by a factor of 256.
X3C: Points in probability space are not equiprobable. ($-3$).
X3D: Failure to cancel common factors in numerator and denominator, as required by the instructions. ($-1$ or $-2$, depending on how many factors were missed)

**Problem 4:**
**a.** $S(n)$ is: If $L$ is a list of length $n$, then `lookup(x,L)` returns `true` if and only if $x$ is on $L$.
**b.** $n = 0$ is the basis.
**c.** If $n = 0$, then the pattern of line (1) matches. `lookup` returns `false`, which is correct because $x$ cannot be on an empty list.
**d.** If $n \geq 0$ and $L$ is of length $n + 1$, then the pattern of line (1) fails and line (2) matches.

If $x = y$ on line (3), then $x$ is surely on $L$, and `lookup` returns `true` as it should.
If $x \neq y$, then $x$ is on $L$ if and only if $x$ is on $ys$. We know that the call on line (4) correctly answers the question whether $x$ is on $ys$, by the inductive hypothesis (since $ys$ is of length $n$).
Thus, the given call to `lookup` correctly answers whether $x$ is on $L$.
Some common errors:
X4A: Not mentioning $n$ in $S(n)$. $(-2)$
X4B: Wrong basis, usually $n = 1$.
The problem is that if you don't choose $n = 0$ you don't prove correctness for the empty list, which is required by the statement of the problem. $(-1)$
X4C: No use of the inductive hypothesis or otherwise failing to create an inductive proof (even if your argument had some validity or intuitive appeal). $(-8)$
X4D: Ignoring the case $x = y$ in (d). $(-2)$

**Problem 5:**
**a.** The correct basis is a single node by itself. This is an odd tree since all interior nodes (all none of them) have an odd number of children, and the tree has one leaf, the node itself. One is an odd number. Many people got tangled up in the definition of odd tree and tried to define more complex bases. Common mistakes were to define the basis as a two node tree with root and single leaf ($-2$ points) or as any tree with root and an odd number of children, all of which are leaves ($-1$ points). The problem with these "bases" is that when you do the induction, you will end up with trees that can be neither handled by the basis case nor further broken down by the inductive case. For example, when you try the induction, you may end up with single leaves as subtrees, which must be dealt with somehow. Thus, the proof will be invalid. Some people also lost points for making basic errors in the definition of a leaf, an odd tree, a subtree, etc.
**b.** Consider any odd tree other than a single node. By the definition of odd tree, the root must have an odd number of subtrees, and each of these must be odd trees, too. Therefore, by the inductive hypothesis, each of these subtrees has an odd number of leaves. The total number of leaves in the tree is just the sum of the number of leaves in the subtrees. Since the sum of an odd number of odd numbers is odd, the entire tree has an odd number of leaves. QED Another correct inductive step considers any tree (not just odd ones) and notes that $S(T)$ is vacuously true when $T$ isn't odd. Minor errors in wording lost you 1–4 points. More serious errors (assuming all subtrees have the same number of leaves, only considering the possibility that the root has 1–3 children, confusing the definition of an odd tree (which tells you that every interior node has an odd number of children) with the inductive hypothesis (which tells you that an odd subtree has an odd number of leaves), etc.) lost additional points.

I (Alan) was disturbed that a signficant number of people were still making fundamental errors in the form of their inductions. Induction is a basic skill you need to know; you will need it again and again, in CS109B, in CS143, CS154, CS161, CS221, etc., etc. Once you get the hang of it, it's easy, and you need to make it so natural that an exam question like this is as easy as falling off a log. If your exam has "NO!!!" or something similar written on the inductive step, I recommend you see a TA or Prof. Ullman during office hours to make sure you understand how to do induction. The major errors in induction included:

- Trying to hand-wave an argument about breaking things down and "continuing until you get to the end." This is not inductive proof, just as writing:
  `fun fib n = if n=1 then 1 else if n-1=1 then 1 else if n-2=1 then ...`
  is not a recursive function. Induction is an **easy** way to take one of these "continue until the end" arguments, and write it down precisely, correctly, and with much less intricate thinking.

- A backward argument in which you take smaller cases where the inductive hypothesis holds and try to "grow" them somehow into larger cases where the inductive hypothesis still holds. We've warned you not to do this since Problem Set 1, so I was worried to see people still trying. (Perhaps it's from being mistaught induction previously, since this error doesn't lead you too far astray when you're doing trivial toy proofs on properties of integers.) I'm not harping on this point just because I'm picky; doing things this wrong way can give you wrong proofs of wrong results, because not everything can be "grown" from the smaller examples. For example, it's easy to generate a wrong "inductive proof" that all strings have even length: (basis) the empty string has length 0, which is even, and (induction) if I concatenate two strings of length less than $n$, by the inductive hypothesis, each string has even length, so the new string also has even length (sum of two even numbers is even).

**Problem 6**

**a.**I (Alan) was not picky about which direction you had which word going:

| | | | | | | |
|---|---|---|---|---|---|---|
| a | 4 | 0 | 1 | 2 | 2 | 3 |
| c | 3 | 0 | 1 | 1 | 2 | 2 |
| b | 2 | 0 | 1 | 1 | 1 | 1 |
| a | 1 | 0 | 0 | 1 | 1 | 1 |
| | 0 | 0 | 0 | 0 | 0 | 0 |
| | | 0 | 1 | 2 | 3 | 4 |
| | | | b | a | c | a |

The left-most column and bottom row are all 0s because the LCS of the empty string with any string is the empty string. The other entries are generated as the maximum of the entry to its left and the entry below if the letters don't match, and as one more than the entry diagonally below and to the left if the letters do match.

**b.** The length is 3, taken from the upper-right corner of the above table.

**c.** The two possible LCSs are `aca` and `bca`. To generate these, start from the upper-right corner and figure out how you could have generated that number. In this case, the 3 at position $(4, 4)$ must have been generated by the 2 at position $(3, 3)$, so any LCS must end with an "a". Similarly, the 2 at position $(3, 3)$ must have been generated by the 1 at position $(2, 2)$, so any LCS must end with "ca". That 1 could have come from either position $(1, 2)$ or $(2, 1)$. If we came from $(1, 2)$, we get an LCS of "bca"; if we came from $(2, 1)$ we get an LCS of "aca".

**Problem 7:**

```
fun philter (_, _, nil) = nil
  | philter (f, x, y::ys) = if f(x, y)
                            then y::philter (f, x, ys)
                            else philter (f, x, ys);
```